

COPYRIGHT

Copying articles from this issue, is only permitted if you mention where they come from, e.g. 'ESGG Vol.1982, N°3, page 3', and only when you are a non-commercial subscriber.

Others must get a written permission from the editor
(address on page 1)

ADVERTISEMENTS

Non-commercial ads are welcome on following conditions:

Format: 1 line = 66 characters or spaces

Max. 6 lines per ad

Price: 3 Dutch guilders for 2 lines

6 Dutch guilders for 4 lines

9 Dutch guilders for 6 lines

Address: send your text to the editor, and at the same time your payment to postal-cheque account 2058443 of ESGG, Utrecht, Holland. (Please mention on the form "Micro's".)

Would you like to receive this paper regularly?

Then please use the subscription form on the last page.

ESGG - Service

The following articles are available for the moment, if you want something, pay on the postal-cheque account 2058443 of ESGG, Utrecht, Holland. Please clearly mention what you want, and don't forget your address. These listed prices are only valid until you receive the next issue of the publication, and for one piece, in guilders

	If you come and get it.	By post.
Cassettes with programs (1-5)	7.50	10.00
Basic Extension (EPROM)	20.00	25.00
Inverse-video print (kit)	7.50	12.50
Single issues of this paper	3.50	4.50

INTERVIEW

Every Sorcerer-user around here know the names "Van Montfort" and "Gielen". The brothers Van Montfort (Harry, Huub and Tony), and Gielen started a (little) firm, called ESAG (stands for "Exidy Sorcerer Application Group"). They are, as far as I know, the only firm only dealing in hard- and software for the Exidy-Sorcerer. That fact makes them deserve of being put in the open.

Already in the beginning of the micro-age, they were obsessed in the possibilities of the micro-processor, and over here, in Holland, among the first users of the TRS80-machine. The television course "Microprocessors 2", and on that occasion, the possibility to obtain a Sorcerer for less then f.2000, made them change and choose our side. Soon they discovered the restrictions and bugs in the "Standard Basic-pac", and for a few Sorcerer users they made the "ROM-Pac simulator". This is a hardware intervention, enabling to write in the addresses from C000 to DFFF. They gave a lecture about it on the HCC-day in Breda (Holland) in 1980. This was the start of the whole thing!

They decided to make a "ROMPAC BUS - System", not finished yet, but that lead them to the development of the (EP)ROM/RAM-Packs, an Eprom-programmer, an I/O-Pack, and a lot (of good) software. This developments were needed as they were not available from EXPERT (the general dealer) or even COMPUTATA (the importer), or they were far to expensive.

The few programs from them on the ESGG-software tapes, date from the period before they had commercial plans. They started selling things when they needed money to continue their hardware developments. Leo Gielen, first helping them in testing the hard- and software, joined the team. Hen imported among other things printer software (GRAFTRAX 80) and all the Arrington products.

ESAG-products:

- an original BASIC (ready but not on the market yet)
- The "Switch-Pac (a mini bus-system in a pack)
- an EPROM-programmer, and eraser
- I/O-pack, EPROM-pack, RAM-pack (with battery back-up!)
- the program "EXPAN" (a monitor expansion)
- the monitor 1.3

etc... Their prices are very low, and they need and deserve our support.

For all information their address: Exidy Sorcerer Application Group

Postbus 6051

NL-6401 SC Heerlen

Holland.

Again: BASIC-EXTENSION

If you want to use the USR-function, without changing the input-vector (as is done in BEXT), you can put in your BASIC-program the following line:

```
10 PO=-4096:PE=-16384:FOR I=PO TO PO+26:POKEI,PEEK(PE):PE=PE+1:NEXT
```

This line initiates a block of variables, used by different USR-routines.

Floor Vogelaar

Answers to your letters:

As mentioned in ESGG 2, our member Martin Staring from Tilburg, Holland, pointed us at an article in the February-issue of Wireless World (page 50). This article tells about Data-recording on audio cassette, very reliable, at 2400 Baud.

After studying the outline, a few differences occur, despite of the similarity on first sight, with the Exidy system.

To decode a digital signal, the time between two "zero-potentials" of the recorded signal, must be measured.

This is done in our Exidy by means of an internal "standard time". So two signals from different sources (one from the computer and one from the recorder) are used, and this (among other things) causes trouble.

In the Wireless World-way, they don't use that standard time signal, but a derivation of the recorded signal itself. So the principle is improved, but is it possible to modify our Sorcerer? First of all, it cannot be done by adding a few wires, or even components.

Second, with all the differences between the cassette recorders, the fase-correction filter must be checked with the recorder used. And third, don't start building the sircuit, if you don't have the required knowledge, a lot of free time, and above all patience; as thar road is paved with failures.

But, once again, the principle is better then the one used in our Sorcerer, and it must be possible to use it.

Let me, and all of us know, if you succeeded.

Aad van Duyvenbode

INPUT ... for asking questions

Send your questions about your hobby to the editor. We don't know everything, but we'll do our best to answer your questions.

Spare-parts program

If you have a lot to do, and one asks you to write for him, a program which sorts a lot of coils by their colour-codes and values in Henry, very often a third person does that for you. The following program is written for an ANCOS-computer, but the author, Etienne François, says it will work on the Exidy too. Try, and if you have usefull modifications, let us publish it!

```
10 REM ERIC's SPARE-PARTS ADMINISTRATION
20 REM A program from Etienne François PA3AAR Landsmeer, Holland
30 REM -----
40 PRINT "Eric's Spare-parts Administration"
50 PRINT "===== ": PRINT
60 X=10
100 DIM P(X)
101 DIM K$(X)
102 DIM W(X)
200 FOR Y=1 TO X
210 INPUT "Spare-part number";P(Y)
220 IF P(Y)=0 THEN 300
230 INPUT "Colour-code";K$(Y)
240 INPUT "Value in Henry";W(Y)
250 NEXT Y
300 M=Y-2
305 PRINT: PRINT "Input list": PRINT
310 GOSUB 7000: REM Print unsorted
320 PRINT "-----"
325 PRINT: PRINT
400 REM Sorted by their spare-part number
401 S=0
410 FOR I=1 TO M
420 IF P(I)<=P(I+1) THEN 470
430 P1=P(I): L$=K$(I): W1=W(I)
440 P(I)=P(I+1): K$(I)=K$(I+1): W(I)=W(I+1)
450 P(I+1)=P1: K$(I+1)=L$: W(I+1)=W1
460 S=1
470 NEXT I
480 IF S=1 THEN 400
490 REM Sorting is done
492 PRINT "Sorted on part-number": PRINT
495 GOSUB 7000
496 PRINT "-----"
```

```

497 PRINT: PRINT
500 REM Sorting on the colour-codes
501 S=0
510 FOR I=1 TO M
520 IF K(I)<=K(I+1) THEN 570
530 P1=P(I): L(K(I): W1=W(I)
540 P(I)=P(I+1): K(I)=K(I+1): W(I)=W(I+1)
550 P(I+1)=P1: K(I+1)=L: W(I+1)=W1
560 S=1
570 NEXT I
580 IF S=1 THEN 500
590 REM Sorting is done
592 PRINT "Sorted by their colour-codes": PRINT
595 GOSUB 7000
596 PRINT "-----"
597 PRINT: PRINT
600 REM Sorting by the values in Henry
601 S=0
610 FOR I=1 TO M
620 IF W(I)<=W(I+1) THEN 670
630 P1=P(I): L(K(I): W1=W(I)
640 P(I)=P(I+1): K(I)=K(I+1): W(I)=W(I+1)
650 P(I+1)=P1: K(I+1)=L: W(I+1)=W1
660 S=1
670 NEXT I
680 IF S=1 THEN 600
690 REM Sorting is done
692 PRINT "Sorted by the values in Henry": PRINT
695 GOSUB 7000
696 PRINT "-----"
697 PRINT: PRINT
700 STOP
7000 PRINT "Spare-part number colour"; TAB(28); "value in Henry"
7001 PRINT "=====
7002 FOR Y=1 TO M+1
7020 PRINT P(Y); TAB(18); K(Y); TAB(28); W(Y)
7090 NEXT Y
7092 RETURN
7095 END

```

PINT

The program PINT is from the brothers Van Montfort, and gives a smoother use of a Centronics printer. The printer works with interrupts, so time can be saved, because while the printer is working, the computer can go on with his calculations.

A character to be printed, goes first to a small buffer, from where it is picked-up after an interrupt. If this buffer is full, output is stopped; if it is empty, the interrupts are stopped. You can start the program with a SE $\beta=0059$.

The hardware connection is rather simple. If you look at the 50-pole expansion connector, you'll see a resistor on top of the board, this is the interrupt-input. It must be connected (e.g. with a test-clip) to the input-bit 7 of the parallel connector of the printer. On the Centronics-printer you'll find there the BUSY-signal. The parallel connector must of course be plugged in at the back of the Sorcerer!

The program

```

0000 ;Program: PINT
0001 ;printer-routine on interrupt-base
0002 ;
E01B 0003 VIDEO EQU 0E01BH
0004 ;
0005      ORG 38H                ;interrupt vector address
0038 F5      0006 RST38  PUSH AF
0039 E5      0007      PUSH HL
003A 2A6600  0008      LD  HL,(INTPON)
003D 2C      0009      INC  L                ;inc. interrupt pointer
003E 226600  0010      LD  (INTPON),HL
0041 26FF    0011      LD  H,0FFH
0043 7E      0012      LD  A,(HL)
0044 2680    0013      LD  H,80H                ;needed for strobe
0046 D3FF    0014      OUT (0FFH),A
0048 E67F    0015      AND  7FH
004A D3FF    0016      OUT (0FFH),A
004C F680    0017      OR   80H
004E D3FF    0018      OUT (0FFH),A
0050 3E      0019      DEFB 3EH
0051 00      0020 VIDPON DEFB 0
0052 BD      0021      CP   L                ;buffer empty?
0053 281F    0022      JR   Z,NOINT- $\beta$  ;yes, disable interrupt
0055 E1      0023      POP  HL
0056 F1      0024      POP  AF

```

0057	FB	0025	EI	;no, enable interrupt
0058	C9	0026	RET	
		0027	;	
0059	CD1BE0	0028	CHROUT CALL VIDEO	;entrypoint for SE 0=59
005C	FE0A	0029	CP 0AH	;neglect linefeeds
005E	C8	0030	RET Z	
005F	F5	0031	PUSH AF	
0060	215100	0032	LD HL,VIDPON	
0063	3E	0033	DEFB 3EH	;skip 1 instruction
0064	FB	0034	WAIT EI	
0065	3E	0035	DEFB 3EH	
0066	00	0036	INTPON DEFB 0	
0067	3D	0037	DEC A	
0068	BE	0038	CP (HL)	;buffer full?
0069	28F9	0039	JR Z, WAIT- ~~g~~	;yes, wait
006B	F3	0040	DI	;no interrupt now
006C	34	0041	INC (HL)	;inc. pointer
006D	6E	0042	LD L, (HL)	
006E	26FF	0043	LD H, 0FFH	
0070	F1	0044	POP AF	
0071	77	0045	LD (HL), A	;in buffer
0072	FB	0046	EI	;free for next interrupt
0073	C9	0047	RET	
		0048	;	
0074	E1	0049	NOINT POP HL	;no more interrupts
0075	F1	0050	POP AF	
0076	C9	0051	RET	

Bugs in the power-on monitor

We recieved the next article already some time ago, but had no room to print it.

The author, Jan Bonsel, from Zeist (Holland) gives us the solution, in order to use the serial ports.

There are two serial I/O-routines in the power-on monitor:

TAPEIN (E2DAH) and TAPOUT (E2EEH). The serial I/O goes with an UART via ports 252, 253 and 254. Thoseports have the following functions:

252	used to send byte by byte
253	tells if the UART is free, in order to send a byte, or at input if a byte is recieved.

This port also gives error-messages.

254

To report speed (300/1200 Baud) and the kind of I/O, tape or RS232C. This port is also used for the keyboard.

At reading the keyboard, first must be given which serial (bank) of keys must be read. But in the used routine (KEYBD), the forgot that port 254 also serves to report the status RS232C/tape and the Baudrate. The routine KEYBD:

```
On address EB26 stands 3E01 LD A,1 ;Look for repeat key
EB28 D3FE OUT KYPORT,A ;Send mask
```

Those two instructions set the two leftmost bits of port 254 on zero, so cassette I/O on 300 Baud. This happens once again on:

```
EB3F 0EFE LD C,KYPORT ;Load KEYBRD POR
EB47 2600 LD H,0 ;Clear section C
EB4D ED61 SLOOP:OUT (C),H;Output section
```

It would be better to take here first the Baudrate-byte from the MWA, and then, combined with the section-number, do the output-command. Also better would be to put the RS232C/cassette status in the baudrate-byte of the MWA. On address EB26 had to be:

```
LD A,(IY+CMTRFG)
OR A,1 ;Add SECTN CTR
OUT KYPORT,A ;Send mask
```

And also on address

```
EB3F LD C,KYPORT (unchanged)
EB47 LD H,0 (ditto)
EB4D SLOOP: LD A,(IY+CMTRFG)
OR A,H ;Add SECTN CTR
OUT (C),A ;Output SECTN NO
```

This fault is the cause that you can not connect a serial printer, or connect two computers with each other serial, unless you don't use the standard keyboard routine of the Sorcerer.

From the redaction: in the monitor 1.3 from Van Montfort, this bug is already corrected.

Who dares ??

In order to exist, this publication needs input from our members, regarding the hard- and software of our Exidy Sorcerer.

We are planning a series of articles about the routines in the monitor and the Basic-pack. Is there anyone willing to start, even in a foreign language, such a series?

Relocating ROM-Pack Basic

This is an article from Bob Stafford, a member of SCUA.

If you use a disk drive with a non-removable controller between C000 and DFFF, and you want to run Rom-Pack-Basic programs, it is useful if you can relocate this Basic.

This method works with both Basic 1.0, and 1.1 versions.

Tested starting addresses are 5000, 6000, A000, A400, A800 and B000.

You must start to move the code to a suitable place in memory.

Then change all the instructions (w.DISAS or an other disassembler) resembling on memory-addresses. Doing this, all the jumps and other data-areas invalid! If you follow these directions, those jumps and areas will be adapted, the addresses mentioned are such, if the program was still between C000 and DFFF.

address	alter
CC0B-CC0D	ENter 00 00 00
D071-D073	ENter 00 00 00
D0C0	ENter 00
CCE2	MOve byte CC by hand
D8D0	also byte CA
D3AB	ENter D1
D4EB	ENter D1
C0C6-C0F5	MOve address table by hand
C1E1-C21C	also
C21D-C231	also with the second and third byte of every three bytegroup
CF2E-CF41	MOve by hand the bytes on CF30, CF33, CF36, CF3A, CF41 when they are in the area C0-DF

It is also necessary to change the code in Basic's cold-start area (C01D-C02F). The easiest way to do so is by ENtering on C01D xx XX (LD HL, XXxx), followed by C3 3F C0 (JP C02F), where XXxx is the address of the Top of Ram you want to use for the Basic stack/stringarea. Of course C02F points at addresses as if the program was still on C000. It is very handy to have a list at hand with the original jump-addresses.

Reading and writing source-files

How it works:

After RUN, you get the menu; we start with the writing of a program, and so we choose a '2' for input. You get a '0' (nul) on the screen, which means you may write. The automatic line-numbering comes not in memory, it is only for the ease of the editor. While writing you can go back with <-- (shift 4) and go on writing after the correction, and then give a CTRL Y. Only when we putted a new END in the program we can end with a CTRL X! It is often easy to put at the end of each line a few spaces, because when you correct afterwards, the new line can become a little longer. The input-routine is based on the new Eprom (BEXT, Basic-Extension, from the ESGG) INKEY-routine I=USR(7). But if you don't have that Eprom:

```
ad 5 POKE 318,195: POKE 320,224
```

```
alter 160 X=INP(9): IF X=0 THEN 160
```

delete line 170.

Tis handy get-routine comes out of "Sorcery Brews".

The program:

```
10 PRINT CHR$(12)
20 PRINT: PRINT: PRINT TAB(15)"** MENU **": PRINT: PRINT
30 PRINT "WPF reading ..... 1": PRINT
40 PRINT "Input ..... 2": PRINT
50 PRINT "Editor ..... 3": PRINT
55 PRINT "STOP ..... 4": PRINT
60 PRINT "Make your choise ";U
70 ON U GOTO 100,150,300,400
100 N=2063: PRINT CHR$(12): A=0: PRINT A;
110 FOR I=0 TO 1: X= PEEK (N): PRINT CHR$(X);
115 IF X=13 THEN PRINT: A=A+1: PRINT A;
120 N=N+1: I=-(X=3): NEXT
130 GOTO 20
150 N=2064: PRINT CHR$(12): A=0: PRINT A;
160 I=USR(7): IF PEEK(-4041)=0 THEN 160
170 X=PEEK(-4041)
180 IF X=13 THEN PRINT: A=A+1: PRINT A;; GOTO 230
190 IF X=25 THEN 260
200 IF X=1 THEN N=N-1: PRINT CHR$(X);: GOTO 160
210 IF X=24 THEN X=3: POKE N,X: GOTO 260
220 PRINT CHR$(X);
230 POKE N,X
240 N=N+1
250 GOTO 160
```

```

260 GOTO 20
300 N=2064: PRINT CHR$(12)
310 INPUT "Line-number ";A
320 FOR K=0 TO A
325 IF K=A THEN PRINT A;; GOTO 160
330 X=PEEK(N)
340 IF X=13 THEN 360
350 N=N+1: GOTO 330
360 N=N+1: NEXT K
400 END

```

TOOLKIT with BEXT (Basic Extention)

There are more jumps to Basic-Warm (DFFA) then mentioned in the previous ESGG publication. You'll find here the needed modifications to be able to use TOOLKIT with the BEXT-Eprom in the pack.

1: LO (not LOG!) TOOLKIT, and alter the following

EN 089E (CR)	6B C0 / (CR)
EN 0A6E (CR)	6B C0 / (CR)
EN 0A93 (CR)	6B C0 / (CR)
EN 0C94 (CR)	6B C0 / (CR)
SE F=54 (CR)	SE X=0E00 (CR) SA(ve) TLKIT 0600 0F3F

2: Toolkit 16, 32 or 48k version:

Load this version with LO (not LOG) and alter:

EN &A6E (CR)	6B C0 / (CR)
EN &A93 (CR)	6B C0 / (CR)
EN &C94 (CR)	6B C0 / (CR)
SE F=54 (CR)	SE X=&880 (CR) SA(ve) TLKIT &600 &DFF

(CR) means give a RETURN. '&' is for 16k='3', for 32k='7', and for 48k='B'.

Floor Vogelaar.

BEXT video-driver

A new development of the always busy ESAG, is the correction of a bug in Basic Extension:

The video-driver used works only with monitor 1.0. With the following alterations the video-driver will work with monitor 1.0 and 1.3

The improvements:

-The CLEAR-key (CTRL L) erases the lower part of the screen, and not the character on address F080H.

- The graphics are not resetted any more.
- CTRL R erases the upper part of the screen. If address F026H is poked on '0' (nul), no crash will follow after CTRL R.
- TAB (CTRL I), moves the cursor 8 places to the right, starting from the absolute left marge!

The program:

ADDR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DC10													FE	20	38	02
DC20	EE	80	F5	F5	C5	D5	E5	01	07	00	21	A5	DC	ED	B1	E2
DC30	F9	E9	21	29	EA	E5	CD	E8	E9	41	10	0C	7B	C6	08	E6
DC40	38	FD	77	6A	FD	70	6B	C9	10	2F	FD	5E	68	FD	56	69
DC50	6B	62	29	29	3E	1D	BC	21	40	00	28	08	19	FD	75	68
DC60	FD	74	69	C9	E5	CD	AB	DC	EB	E5	21	40	07	ED	52	E3
DC70	C1	19	EB	E1	19	ED	B0	18	05	10	10	CD	AB	DC	EB	7A
DC80	B3	C8	36	20	23	1B	CB	5C	28	F5	C9	10	06	CD	9C	DC
DC90	19	18	EC	10	07	ED	5B	38	F0	C3	F0	DD	CD	AB	DC	CD
DCA0	5D	DC	AF	18	9C	11	07	0C	12	0A	09	3A	26	F0	11	80
DCB0	F0	E6	0F	6F	63	06	06	29	10	FD	C9					

SYSTEM 3 with BEXT

LO SYS3 (not LOG!) and make the following alterations:

EN 1C98 (CR)	6B C0 / (CR)	
EN 1D23 (CR)	6B C0 / (CR)	
SE F=30	SE X=0F00	SA SYS3A 0F00 1F4F

Floor Vogelaar

Latest News

The somewhere else published modification of BEXT, from Van Montfort, together with a few other minor shanges, will be in the supplied BEXT-Eproms, the version is called BEXT8.

The only change in using it, will be that CTRL is replaced by CTRL 1, to use the linker.

The way to update the older version, BEXT3, into BEXT8, will be in one of the following publications. Also the memory-map of the Eproms, promised some time ago will be printed.
